DOCUMENT RESUME

ED 234 757 IR 010 823

AUTHOR Bryan, John T.; Felty, J. Michael
TITLE The PCP SYS IV Management System: Technical Overview.
INSTITUTION Illinois Univ., Urbana. Computer-Based Education Research Lab.
REPORT NO CERL-E-28
PUB DATE May 83
NOTE 18p.; For related documents, see IR 010 822-824.
PUB TYPE Reports - Descriptive (141)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; *Computer Managed Instruction; *Computer Programs; Curriculum Design; *Design Requirements; Instructional Systems; Program Descriptions; *Programing Languages
IDENTIFIERS Instructor Logic Programing Language; Interactive Systems; *PLATO IV

ABSTRACT

This paper is the second in a three-part series on SYS IV, an instructional management system developed by the PLATO Curriculum Project (PCP) that incorporates some of the features of the previous SYS systems. Focus is primarily on the new features that make the system unique. Following a brief background section explaining the purpose of SYS IV, primarily technical issues are addressed. Specific features described are the SYS IV files and files structures, which include the system catalog file, the subfiles files, and system files list. The programing language Instructor Logic (IL), whose specific purpose is to provide a curriculum designer with flexibility in curriculum structure, is explained, and all IL commands are listed with a brief description of their use and purpose. Additional functions described are the router, group editor, catalog editor, implementation scan utility, and the testing package. Final comments address the future of SYS IV. An appendix provides a sample of an IL program from the PCP Reading Comprehension material. (LMM)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.
Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

Computer-based Education Research Laboratory

CERL

ED234757

THE PCP SYS IV MANAGEMENT SYSTEM: TECHNICAL OVERVIEW

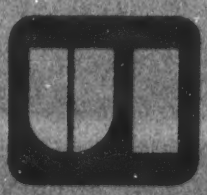
JOHN T. BRYAN

J. MICHAEL FELTY

CERL REPORT E-28

MAY 1983

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
D. Bitzer
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."



University of Illinois at Urbana-Champaign

IR010823

Copyright © May 1983
by the Board of Trustees, University of Illinois

All rights reserved. No part of this book may be reproduced in any form or by any means without permission in writing from the authors.

The PLATO® system is a development of the University of Illinois.

PLATO® is a service mark of Control Data Corporation.

Acknowledgement

The PCP SYS IV Computer-based Management System was developed by Martin A. Siegel, Project Director; J. Michael Felty; John T. Bryan; Stephen M. Alessi; Lindsay P. Reichmann; Walter G. Brooks; David A. Fuller; John B. Gilpin; and Peter K. Zelchenko.

This paper is the second in a series on SYS IV. In the other two papers, Martin A. Siegel and Dennis M. Davis present an educational overview, and Dorothy Z. Silver and Elizabeth J. Clapp outline an implementation and training overview.

THE PCP SYS IV MANAGEMENT SYSTEM: TECHNICAL OVERVIEW

John T. Bryan
J. Michael Felty

PLATO Education Group
Computer-based Education Research Laboratory
University of Illinois at Urbana-Champaign

SYS IV is an instructional management system that incorporates many innovative and powerful features. SYS IV, as its name implies, is the fourth in a series of instructional management systems developed by the PLATO Curriculum Project (PCP) each of which has been more sophisticated and powerful than the last. Some of the features of SYS IV existed in previous systems; some are new. In this paper we will concentrate mostly on the features that are new because these features, we feel, make the system unique. We will also address predominantly technical issues after providing a brief background explaining the purpose for SYS IV.

Because SYS IV is the product of experience gained in developing and using previous systems, it is useful to elaborate somewhat on what the previous systems had to offer and what improvement a new design could provide. SYS III offered all of the features needed by a management system; communications between student and instructor, integrated curricula with data keeping, and the ability for the instructor to both monitor and control student progress -- but a fundamental flexibility was missing. After SYS III had been in use for some time, we decided to add computer-managed testing to the package, and this proved possible, although cumbersome. We realized later that testing needed to be made a more integral part of the system in order to use the results more effectively. For example, if a student is working on mathematics and is missing only questions on multiplying fractions, it would be desirable for that student to work only (or perhaps mostly) on lessons involving the multiplying of fractions, and not spend his time on other matters he already knows. For the populations served by the PCP, who often enter the program with diverse backgrounds and capabilities, this can be particularly important. Because of the way testing was connected to SYS III, however, this kind of intra-curriculum communication was difficult and we designed a more integrated testing package. It was clear that there were other changes desired for SYS III as well, and we began discussing how best to achieve these additional needs. Soon after discussion ended and a design was agreed upon additional concerns arose, and a new design was mandatory. Built in curriculum structures simply could not provide sufficient flexibility to accommodate proposed curriculum designs.

The chief source of SYS IV's flexibility is a programming language called "Instructor Logic" (IL). This also provides an easy means to expand the system, because one need only add a new command to the IL

language to implement some new feature. A curriculum is designed and then implemented in IL with whatever special features are necessary to that particular curriculum, utilizing a fairly diverse set of options. A curriculum might consist of many "units" of instruction, or a single lesson. Because a single lesson doesn't really constitute a "curriculum," they are often referred to as "topics." Within a topic may be instructional lessons, tests, assignments, and instructor decision points, all as individualized or as fixed as desired. How IL provides this capability will be discussed later.

Another departure from SYS III was an alteration of the data-keeping structures. In concept, SYS IV data is similar in some ways to SYS III, but a fundamental characteristic of SYS III data was its permanence. For SYS IV, it was decided that information has a "half-life" after which its usefulness is diminished. The critical problem was determining what portion of the data remains useful, and for how long. The rationale and means for reducing data bulk while maintaining data usefulness are, we feel, innovative.

Technical Background

The SYS IV Student Routing and Management System is written in the TUTOR language on the PLATO computer system. In many ways, the capabilities and restrictions of the PLATO system influenced the design and implementation of the SYS IV package. However, many specialized structures within the PLATO system also reduce the need to develop complex tools before beginning construction of the system, among these: a system-level distinction in user types, dynamic file structures keyed by a name (namesets), a sophisticated protocol for routing a student into instructional lessons (routers), the ability to establish a buffer of memory that is shared by more than one user (common), files for communication among groups of PLATO users (notefiles), and so forth. We have used most of the available features of the PLATO system in developing SYS IV.

SYS IV also uses one feature normally restricted to system programs called "SHIFT-STOP protection," which prevents a program from being interrupted by a SHIFT-STOP keypress as is usually the case. This protection is applied only to the "router" where critical data-keeping is performed. SYS IV also includes several maintenance programs which can be run manually but which are much more appropriately run periodically and automatically. The "CERL" PLATO system now has a feature that makes it possible to sign-in a station that is unused. We use this feature to run maintenance programs daily or weekly.

Because of the fundamental design philosophy used with SYS IV, that the system must be easy to expand, most of the file structures and programs have space deliberately left unused for future expansion.

Because most features of the PLATO system are well documented, in this paper we will concentrate on how SYS IV uses the capabilities of the

PLATO system, and notably how SYS IV uses PLATO file types. There are several varieties of PLATO files: lesson files (where programs and lessons are written), datasets (a collection of fixed-size records), namesets (a collection of fixed-size records that can be dynamically allocated to "names"), groups (a collection of names and information used by the PLATO system to determine user identity and to maintain data on each individual user of the PLATO system), and several others. SYS IV uses these files in many different ways.

SYS IV Files and File Structures

SYS IV requires that each student and instructor have a signon in a "group" file. Each signon in a group must be entered into a "section" within that group (sections are lettered "A" through "Z"), which provides an easy way for an instructor to manage several sets of students, or for several instructors to use the same group file without interfering with each other. The group file name must be entered into a list of groups that are in some way associated with each other, and these groups share a common "catalog." This collection of groups is a "SYS IV system," and each system has a name. For example, we currently have a system named "PCP," which consists of 20 groups. The system has a "System Catalog" file (a nameset); a "Subfiles File," which contains three "subfiles": the "System Common," a file list (usually called a "leslist," for lesson list), and an access list (which controls which signons have permission to perform certain actions within the system); a "Daily Usage Summary," which keeps a record of statistics on the usage of signons during each day; and several other files with little relevance to this discussion. In addition, each system may contain up to 50 group files. Each group file may have several associated files: the "Log File," the "Archive File," the "Summary File," and the "TABE Test File."

The System Catalog File contains the catalog of topics that are available to students in the system. However, two restrictions are placed on a student's use of a topic:

1. The topic must be placed into the "Group Catalog." The Group Catalog contains only a list of pointers to topics in the system catalog and is stored in the group file. This facility permits some high level control of what topics are available to what groups, and simplifies the task of the instructor by providing him with a useful subset of all available topics.

2. The instructor must assign the topic from the Group Catalog to the student. A student may have up to nine topics assigned at one time.

The System Catalog allocates one name for a directory, which is a list of areas. With each area is a list of topic numbers included in that area. For each topic in the catalog, a name is allocated with several records (at least) attached to it to hold a large number of pointers and

initial condition variables, most of which are required by the IL executor. A record is included to hold a textual description of the topic, which is used by instructors to determine the usefulness of a topic for their students, or to provide information to the instructor about the functioning of the topic. Finally, each unit of instruction is allocated one or more records for its IL program (more detail is provided about IL later in this paper). Topics are always referred to internally by their topic number which, for the current implementation of SYS IV, is in the range of 1-4095. Given a topic number, the title for that topic is found easily by finding the title number that is the negative of the topic number (for example, topic #807 has title number -807).

The Subfiles File contains three "subfiles." The "System Common" and the "System Files List" are the most important, while the "Access List" is simply a convenience that is provided by the PLATO system, since other access controlling mechanisms also exist within SYS IV. The System Common is a normal PLATO "permanent common," which means that when no one is using the SYS IV system, the contents of the common are stored on disk. Whenever anyone is using the system, however, the common is copied into the PLATO system Extended Memory and is shared by all users of the system. Keeping data in the common makes accessing it very fast and simple, but there is a definite limit on the size of this common (8000 words, each word = 60 bits). The common is divided into areas:

1. General system information is kept in the first 300 words, including information for the Daily Usage Summary (for which day data is being collected, what the next record is in the file, and so forth). Some of this space is unused, saved for future development needs.

2. The Daily Usage Summary information for the current day is accumulated in the next 150 words (this amounts to 3 words for each group in the system). This section of the common is copied to the Daily Usage Summary File once per day and reinitialized.

3. The Group and Section Flags are stored in the next 50 words. These flags determine whether data is kept for each group in the system, and for each section in each group. Two additional flags determine if the group is turned off (which prevents anyone in that group from signing into the PLATO system), and if the group is "secure" (a mechanism used to prevent users of a group from signing into the PLATO system away from their own site).

4. The next 320 words are reserved as the Title Overlay Area, which is reserved for future use.

5. The last section of the common is the Titles Area, and may have any length as long as the total length of the common does not

exceed 8000 words (giving this area a maximum of 7180 words). Each title stored in this area requires 5 words, providing a maximum of 1436 titles, and each has a maximum of 47 characters (however, the title must display on a PLATO screen in 40 character spaces or less). The remaining 18 bits in each 5-word title buffer contain a count of how many times the title has been referenced during the current day, and the title number. Titles are stored in ascending order by title number, making it possible to use fast searching algorithms to locate a particular title given its number. The total number of titles currently stored is kept in the General System Information area.

Likewise, the System Files List is divided into several sections:

1. The System Catalog File name is held in the first slot of the files list.

2. The next 50 slots contain the names of the group files.

3. The next 50 slots contain the names of the Log Files associated with the groups.

4. The next 50 slots contain the names of the Archive Files associated with the groups.

5. The next 50 slots contain the names of the Summary Files associated with the groups.

6. The next 50 slots contain the names of the TABE Test Files associated with the groups.

7. The next 50 slots contain various system files, and room for expansion:

   a. The System Daily Usage Summary File

   b. The System HELP Lesson (the lesson that a student is routed to when the HELP key is pressed in the router, and which may be customized for each SYS IV system).

   c. The System Sequencer File

   d. The System-wide Notesfile (a PLATO notesfile used for communications among all groups in the system).

   e. The System Maintenance Log (used with the maintenance program to maintain a log of what maintenance activity has been necessary).

   f. The ISU Prints File, the ISU Binary File, and the ISU Lists File.

   g. The System TABE Prints File

The rest of the files list contains the file names of TUTOR lessons used in the various topics. No lesson name appears more than once in this list. Since PLATO "leslists" have a maximum size of 2400 file names and the first 301 slots are used for system files, 2099 lessons may be referenced by the topics stored in the System Catalog.

The Daily Usage Summary provides an overview to instructors or other interested persons concerning the daily trends in the use of SYS IV. Basically, how many different students signed into the PLATO system, how long they spent working on topics or doing other activities, how many instructors signed into the system and how long they spent working on various activities, and other cumulative data for a single day is kept.

By far the most complicated of the SYS IV data structures is what may globally be referred to as Student Log Data. The student log, in the abstract, contains all the data the instructor needs to determine what the student has done and is doing, and also allows the curriculum designer to evaluate the effectiveness of his curriculum. In past systems, the student log was a permanent structure that continued to grow as long as the system continued to be used. In practice, this expansion cannot be permitted to continue indefinitely, as disk storage space is finite. For SYS IV, a compromise had to be reached between maintaining the data that actually is necessary for evaluation and instructor effectiveness, while reducing the long-term data-keeping to an effective zero over time.

The scheme employed by SYS IV is simple in theory: the data is kept in one of four different states:

1. For each topic assigned to each student, one record is allocated to hold detailed information on the most recent unit of instruction. Each time a student moves to a new unit, this data is sent to 2. This data record (which is referred to as a "Log" entry) contains three parts:

    a. The Unit Summary. Information such as minutes in the unit, number of different days in the unit, starting and ending dates, test scores and dates, and so forth, are kept in this part of the data record.

    b. The Lesson Summary. Information about each lesson the student worked on in the unit is kept here (the lesson title number, how many minutes spent, whether the student completed the lesson or not).

    c. The Test Log. Detailed information about the last test a student worked on in the unit is kept here. Each question, along with the student's response and the correct response, is available from this log.

2. For each topic assigned to each student, one record is allocated to hold a compressed version of the current unit data

record. The third section (the Test Log) is discarded (which reduces the size of the data by half), so that the Unit Summary and the Lesson Summary may be kept for two additional units in a single record. These two entries are referred to as "Archive" entries. When a student completes a unit and both halves of the Archive record are occupied, the older unit is sent to 3.

3. Before being stored in this section, the Lesson Summary section is removed. This reduces the total data being kept to less than one-fifth of its already reduced size, allowing 10 of these Summary entries to fit in one data record. Data records are allocated in the Summary file as long as the student continues to complete units. When the student is deleted from his group, this data is considered to be type 4.

4. The last phase of a student's data begins when he is deleted. A special entry is inserted into his Summary data containing information about his time on PLATO (how many minutes spent overall, how many different days he worked on PLATO, when he was created and deleted, and so forth). For this student, there is no other record of his individual existence anywhere in the system so, when all analysis of this data summary is completed, a single operation removes all references to this individual student, thus freeing his data space for future students.

As you can see from this process, the trend is toward ever-decreasing amounts of data, but with no real work involved in the compression. We were fortunate in that we could do our compression by simply throwing away sections, but the principle of data compression is still useful even if manipulation of the data or structure is necessary. The data the instructor needs is always available, and the data for curriculum evaluation is contained in the summary entry that is kept until discarded. Using standard procedures, the data kept on deleted students may remain for as little as 24 hours or could conceivably be kept forever (although this defeats the purpose of having the data "evaporating" somewhat).

SYS IV separates the various kinds of student log data (Log, Archive, Active Summary, and Deleted Summary) into three files (the Log File, the Archive File, and the Summary File) in order to make it simple for a program to find the data it wants and to interpret the data it finds, and because a very large group file could easily overrun the maximum size for a log file.

Instructor Logic

Instructor Logic (IL) is a programming language whose specific purpose is to provide a curriculum designer with the widest possible flexibility in curriculum structure. Various attempts have been made over the years

to provide menu-driven curriculum specifications, but, just as with menu-driven lesson specifications, the result is often more complicated than having written a program to do the same thing, or else the range of possibilities is sadly limited. For our own needs, we would have been perfectly content to write our curriculum structures in TUTOR (the programming language normally used on PLATO), but many obstacles stood in our way, not the least of which was the necessity for a whole file for each topic (many of which consist of only a single lesson) or some complicated scheme for using one file for more than one topic. Thus, Instructor Logic.

IL is modelled closely after the TUTOR language because people familiar with PLATO are likely to be familiar with TUTOR, and because our own programming staff all work in TUTOR daily. Thus, the language consists of a group of "commands" that are sometimes followed by "tags" and comments. The tags are generally order-dependent and sometimes optional. The overall structure is divided into "units" which ordinarily follow each other in sequence. Commands are available to send a student to a lesson or a test, to do calculations, to check conditions and perform conditional operations, to stop the student until the instructor makes a decision, and so forth. The expression-evaluation scheme was designed to mimic the functioning of TUTOR as closely as was feasible (the only features left out are exponentiation, and user- and system-defined functions).

A list of all IL commands is included here, each with a brief description of its use and purpose. No detailed language specification exists at this time, but a great deal of information about writing and using IL is available on the CERL PLATO system in lesson "ilaids."

UNIT — This command specifies the beginning of a new unit of instruction. Within each unit a maximum of 256 words of compiled IL commands may be stored, and each IL command takes one word with only a few exceptions.

INDEX — Many times it is desirable to offer a student more than one lesson to work on. The INDEX command and its associated commands (EXIT and CONTROL) allow many different approaches to offering lessons simultaneously.

ENDINDEX — Used to mark the end of an INDEX.

EXIT — This command has two forms. The first form is a conditional or unconditional exit from within an index. The second form allows the student to choose when to leave the index (allowing him to review lessons as long as he feels necessary, for example).

CONTROL — Within an index, this command either places an asterisk by a lesson when the student's choices for the index are plotted (usually used to indicate to the student that he has completed that lesson) or to remove it from the index (to

prevent the student from working in that lesson), or both. It can also remove an asterisk or return a lesson to the index.

BRANCH — This command permits conditional or unconditional branching within the unit or between units. This command has a variety of forms, and some other commands (for example, IF and LOOP) are compiled into BRANCH commands to conserve space, and because they all perform basically the same function.

CALL — This command invokes an IL subroutine, which is much like a unit but begins with BEGIN.

RETURN — This command causes a conditional or unconditional return from a subroutine invoked with the CALL command. A subroutine has an automatic RETURN at its end.

BEGIN — Used to mark the beginning of a subroutine the same way UNIT marks the beginning of a unit.

IF — The IF command, along with ELSEIF, ELSE and ENDIF, provides a general conditional structure virtually identical with TUTOR's.

LOOP — The LOOP, RELOOP, OUTLOOP and ENDLOOP commands, provide a general iterative structure also virtually the same as TUTOR's.

LESSON — The LESSON command sends the student to a TUTOR lesson with many possible options concerning completion, parameters to be sent to the lesson, how to manage the lesson's intermediate status, and so forth. This command is the general interface between IL and TUTOR and is probably the most complex of the IL commands. This command also specifies the title that the student and instructor should be shown for the lesson (and, because of the way IL works, the same lesson may have more than one title in different places, if desired, making lessons that do many different things depending on certain parameters easily supported in SYS IV).

ASSIGN — Used to send the student to a TUTOR lesson much as LESSON does, but there are two differences: the lesson is not entered into the student's log, and unless the IL specifies otherwise, the student will be able to review this lesson at any time. This feature is often used to specify an assignment for the student to keep in mind while he does his PLATO work.

TEST — The TEST command is used to send the student to a test. SYS IV has two built-in testing packages: one is quite general for implementing many different sorts of tests, and the other is for administering the Test of Adult Basic Education

(TABE). The TEST command may also be used to send the student to a lesson that is to be regarded as a test, using all of the features available with the LESSON command.

STOP — This command is used to specify that the topic is completed.

WAIT — This command stops the student and sends a message to his instructor. The instructor must make a decision from up to six available choices specified by the WAIT command before the student may continue.

MESSAGE — A message, selected from a pool of available messages, is shown to the instructor when he looks at the student's records.

TRACK — Used to define an interface between IL, the instructor, and the general testing package for controlling precisely the activities a student will work on.

MAP — This command is basically a programming convenience, and is used for setting up an INDEX structure in accordance with the current settings of the interface established with the TRACK command.

TIMER — The TIMER command establishes a time limit for the student in the next lesson he enters. If he is still in the lesson after that time, the system "pretends" that the student pressed SHIFT-STOP to leave the lesson.

CALC — Provides a general calculating ability.

ZERO — Provides a fast means to set a group of registers to 0.

DEFINE — This command provides the ability to declare named constants, variables, and segments consistent with TUTOR.

DECLARE — This command is used to declare the number of Permanent Registers needed by the topic.

Comments, as in TUTOR, may be specified by placing two dollar signs ($$) before the beginning of the comment, or any line with an asterisk in the first character position is considered to be a comment line. In IL, we also have a Topic Description, and it is generated by placing an equal sign (=) in the first character position of each line of the description.

IL programs have a very limited "variable space," unlike TUTOR. In TUTOR, 150 student variables are always available to every program and, as long as that program is running, those variables are preserved. For IL, information, status, and variables for up to nine topics must all be kept in 64 variables (words). Each IL topic gets a group of special "registers" with special meanings (for example, the register CTIME

contains the number of minutes the student has spent in the topic, and DONE tells whether or not the last lesson was completed). There are also three registers which are available to the programmer for whatever purpose he chooses (called "B1", "B2" and "B3") but with limited range of values (-63 to 63 only), and up to 10 "whole-word" registers (called "X1", "X2" ... "X10") declared with the DECLARE command. However, since the pool of whole-word registers contains only 27 per student, it is essential that each topic use as few as possible to avoid problems with running out of registers for a student's topics (since a student may have up to 9 topics, if a topic uses 3 or fewer registers it is relatively "safe"). In practice, this is not too difficult. Of course, using the DEFINE command, it is possible to give any register a new name or to break it into pieces for more compact data storage.

To compile IL programs a utility was developed and (appropriately) called the "Instructor Logic Compiler." It, as with the rest of the SYS IV package, is written in TUTOR. The IL compiler performs the task of converting the typed-in IL commands to the internal "compiled" format (which usually puts one command in a single 60-bit word), and which also converts all expressions into an internal expression-evaluation format that resembles a machine code for a 5-bit processor. The compiled IL program is stored in the System Catalog file, as mentioned before, and is executed by the SYS IV router when the student chooses a topic. An IL Executor program has also been developed to permit the programmer working on IL to test it in a controlled environment.

Other Aspects of SYS IV

SYS IV may be divided into sections in several different ways, and so far in this paper we have looked at File Structures and Instructor Logic. These two aspects of SYS IV are significant throughout the system, but they are not the whole package.

The Router performs several functions. First, it keeps data on student and instructor use of the system. Second, it executes IL to route the student through his various tests, lessons, assignments, and other activities. Third, it gives instructors (and students, to a lesser extent) access to various utilities (such as notes to and from students), particularly the group editor for instructors.

The Group Editor (or Records Editor) gives an instructor the capability to create and destroy signons, both students and lower level instructors. It also provides the ability to assign and delete topics, to handle WAIT states initiated by IL, to transfer students among sections, and so forth. Most of the options available to instructors are found in the Group Editor.

The Catalog Editors allow a system-level user (usually not an instructor but someone with responsibility to maintain and operate a SYS IV system) to create, destroy, rename, compile or recompile, test, or otherwise manipulate topics in the System Catalog, or to insert, remove or reorganize topics in the Group Catalogs.

The Implementation Scan Utility (ISU) allows instructors and system-level users to scan any subset of the users within a system and format printouts or on-line lists of that subset or statistics extracted from that subset. A simple programming language provides virtually unlimited options in selecting subsets of users and manipulations performed on them, but a "user-friendly" front-end program is also available to permit easier access to the options of the ISU.

The Testing Package permits the development and adminstration of tests in many different formats including multiple choice, matching, passage reading, and several others.

There are also other utility programs, mostly accessed via the System Index, for handling maintenance, system control, and other occasionally necessary options. SYS IV is now comprised of approximately 30 TUTOR programs totalling close to 150 PLATO disk parts (about 3,360 000 characters worth of file space), more than half of which are required by the everyday operation of the system (router, group editor, testing package, and so forth) and the rest being various utilities needed only occasionally (catalog editors, IL compiler, etc.) or debugging routines not needed ordinarly except during development of new features.

The Future of SYS IV

Work is underway to convert SYS IV to operate in a new environment, the CERL Cluster PLATO system, which runs a somewhat modified version of MicroTUTOR within a terminal containing a microprocessor but connected to a powerful minicomputer and network for quick access to programs. Although the work is barely beyond the planning phase, it appears that there will be little difficulty in making the move into the microcomputer environment, and we feel that the prospects for moving SYS IV to other systems are good.

We have presented in this paper an overview of the technical aspects of SYS IV rather than a detailed look at the PLATO implementation because we hope that some portion of our work may be applicable to other environments. The concept of the self-condensing Student Log, and the use of a programming language as a student routing mechanism, we have found to be powerful and useful in developing a management system which meets our needs. We are hopeful that the needs of others may also be satisfied by SYS IV or similar management systems.

Appendix A: Sample of Instructor Logic Program

What follows is part of a sample unit from the PCP Reading Comprehension

```
unit    4,log=on
*--
track   a
        c
        d
        e,lesson,Pacer Passages V
        f,lesson,Pacer Passages VI
*--
call    preunit
*--
if      [flag(intro) = off]
.       assign  introrc
.       assign
.       calc    flag(intro)  on
endif
*--
Opre
test    pretest = xreadt
*--
branch  [(cstatus $mask$ o3) < 2],unit5,x       $$ passed?
call    postpre
*--
calc    t1  [flag(pretest)=on] $$ previous pretests?
        flag(pretest)  on      $$ mark pretest given
*--
if      [t1]
.       wait    x1;prefail;instruct,posttest,(instenab)
else
.       wait    x1;prefail;instruct,posttest,(instenab),(pretest)
endif
*--
branch  [x1-2],x,Opost,x,Opre
*--
Oless   flag(lessons)  on              $$ mark instr. given
*--
if      [track("z") > 1]                $$ offline not done?
.       assign  introrc(unit)           $$ give assignment
endif
*--
zero    x1,x2,x3,x4
*--
calc    flag(indx)  off
*--
index   indexrg
.       calc    temp  indexrg $mask$ cmask
.       map     b,e,f
.       calc    indexrg  indexrg $union$ temp
```

```
*--
lesson  reading2 [x1]/preread;Reading Comprehension II;
        (zscore≥21)
.       calc    rscore  zscore
.       if      [done]
.       .       control reading2;mark
.       endif
*--
lesson  pacer05 [x2]/prepace;Pacer Passages V;(zdone<0)
.       if      [done]
.       .       control pacer05;mark,off
.       .       control pacer06;clear,on
.       .       zero    pscore,x2
.       else
.       .       calc    pscore  zscore
.       endif
*--
lesson  pacer06 [x2]/prepace;Pacer Passages VI;(zdone<0)
.       calc    pscore  zscore
.       if      [done]
.       .       control pacer06;mark
.       endif
endindex
*--
```